Metode Bagging Sebagai Aplikasi Algoritma Divide and Conquer Dalam Pembelajaran Mesin

Rava Maulana - 13521149
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
13521149@std.stei.itb.ac.id

***Abstrak*—Metode *bagging* merupakan teknik *ensemble learning* yang populer dalam bidang pembelajaran mesin. Metode ini merupakan aplikasi dari algoritma *divide and conquer*. Algoritma *divide and conquer* bekerja dengan memecah sebuah persoalan kompleks menjadi subpersoalan yang lebih sederhana. Metode *bagging* sendiri bekerja dengan membagi dataset menjadi beberapa subset. Pembagian tersebut menambah variansi dan memberikan aspek-aspek baru pada data. Pada makalah ini, akan dilakukan pengujian terhadap efektivitas metode *bagging* sebagai aplikasi algoritma *divide and conquer* dalam meningkatkan performa model.**

Kata kunci—pembelajaran mesin, divide and conquer, ensemble learning, metode bagging

I. PENDAHULUAN

Pembelajaran mesin merupakan sebuah revolusi teknologi di bidang kecerdasan buatan yang memberikan berbagai perubahan di bidang industri. Pembelajaran mesin memungkinkan sebuah komputer untuk belajar dari data dan secara otomatis meningkatkan performa dirinya tanpa harus diprogram secara eksplisit. Hal tersebut membuka berbagai macam kemungkinan pengembangan yang dapat dilakukan di bidang teknologi. Perubahan-perubahan di beberapa bidang seperti *computer vision, natural language processing,* dan analisis data terus berkembang dengan cepat.

Semakin mudahnya data untuk diakses, serta semakin meningkatnya kecepatan komputasi memberikan dasar yang kuat bagi pembelajaran mesin untuk berkembang dengan cepat. Dengan semakin banyaknya data yang dapat diakses, metode pengambilan keputusan berbasis analisis data manual menjadi kurang efisien. Hal tersebut disebabkan oleh kemampuan komputer yang dapat melakukan proses komputasi dengan lebih cepat. Algoritma pembelajaran mesin memiliki kemampuan untuk mengambil informasi, menemukan pola yang sulit dikenali, dan membuat prediksi yang akurat dari data yang diberikan.

Salah satu kunci dalam perkembangan bidang pembelajaran mesin sekarang ini adalah pembelajaran mendalam. Pembelajaran mendalam memperkenalkan sebuah model *Artificial Neural Network* (ANN) yang terdiri dari beberapa *hidden layer*. Model pembelajaran mendalam mampu menangkap pola-pola kompleks pada data serta keterhubungannya antar satu sama lain. Hal tersebut memberikan kemajuan di bidang *speech recognition, natural language processing,* dan *autonomous driving*. Algoritma pembelajaran mendalam seperti *Convolutional Neural Network* (CNN) dan *Recurrent Neural Network* (RNN) memiliki performa yang lebih baik daripada model pembelajaran mesin tradisional dalam memberikan prediksi yang akurat.

Algoritma *divide and conquer* memberikan kemampuan tambahan bagi sebuah model pembelajaran mesin. Algoritma *divide and conquer* sendiri sudah dikenal sebagai teknik pemecahan masalah yang efisien dalam memecahkan berbagai macam persoalan. *Divide and conquer* memecah sebuah persoalan kompleks menjadi subpersoalan yang lebih sederhana, memecahkan subpersoalan tersebut secara independen, lalu menggabungkan solusi dari masing-masing subpersoalan menjadi solusi dari persoalan awal.

Sebuah model pembelajaran mesin dapat memperoleh beberapa manfaat dari penggunaan algoritma *divide and conquer*. Teknik memecah sebuah persoalan menjadi subpersoalan yang lebih kecil (*divide*) memungkinkan pelatihan sebuah model dilakukan secara paralel. Komputasi paralel sekarang ini sudah mengalami berbagai kemajuan dengan memanfaatkan kemampuan *multithreading* sebuah komputer. Tahap pelatihan sebuah model memakan waktu yang jauh lebih lama daripada proses komputasi biasa. Proses pelatihan model secara paralel dapat mengurangi waktu yang dibutuhkan untuk melatih sebuah model secara signifikan. Hal tersebut dapat meningkatkan skalabilitas sebuah model dalam menangani dataset yang lebih besar.

Algoritma *divide and conquer* banyak diterapkan dalam bidang pembelajaran mesin. Teknik *hierarchical clustering* membagi sebuah dataset menjadi kluster-kluster kecil secara rekursif dengan tujuan menemukan struktur pengelompokkan data yang natural untuk sebuah dataset. Algoritma-algoritma pelatihan berbasis pohon keputusan seperti CART dan *Random Forest* menggunakan *divide and conquer* dalam membagi sebuah *instance* berdasarkan nilai dari fitur-fiturnya. Metode *bagging* memanfaatkan proses *divide and conquer* untuk melatih model yang lebih independen sehingga dapat menghasilkan prediksi yang lebih akurat.

II. LANDASAN TEORI

A. Algoritma Divide and Conquer

Divide and conquer merupakan sebuah algoritma dasar dalam bidang matematika dan ilmu komputer. *Divide and conquer* bekerja dengan membagi sebuah persoalan yang kompleks menjadi beberapa persoalan yang lebih sederhana, menyelesaikannya secara rekursif, dan menggabungkan solusinya menjadi sebuah solusi akhir.

Tahap pertama dari algoritma ini membagi sebuah persoalan menjadi subpersoalan yang lebih kecil. Pembagian ini dapat dilakukan secara rekursif hingga persoalan menjadi cukup sederhana untuk langsung diselesaikan. Tahap pembagian ini bertujuan untuk mereduksi kompleksitas dari persoalan sehingga membuatnya menjadi lebih mudah untuk diselesaikan.

Tahap berikutnya adalah tahap penyelesaian subpersoalan. Solusi dari subpersoalan yang sudah sederhana diperoleh menggunakan algoritma yang sama. Proses rekursif dari algoritma akan berhenti ketika mencapai kasus basis. Tahap ini sendiri menentukan efisiensi dari algoritma dan berpengaruh pada kompleksitas waktu secara keseluruhan.

Tahap terkahir adalah tahap penggabungan solusi dari persoalan-persoalan sederhana. Penggabungan tersebut dapat dilakukan dengan berbagai cara seperti *merging*, *sorting*, atau operasi lainnya tergantung dengan jenis persoalan yang diselesaikan. Tahap ini memegang peranan penting dalam menghasilkan solusi akhir karena harus dipastikan bahwa subpersoalan sudah diselesaikan dengan benar dan saat digabung akan menghasilkan solusi yang benar untuk persoalan awal.

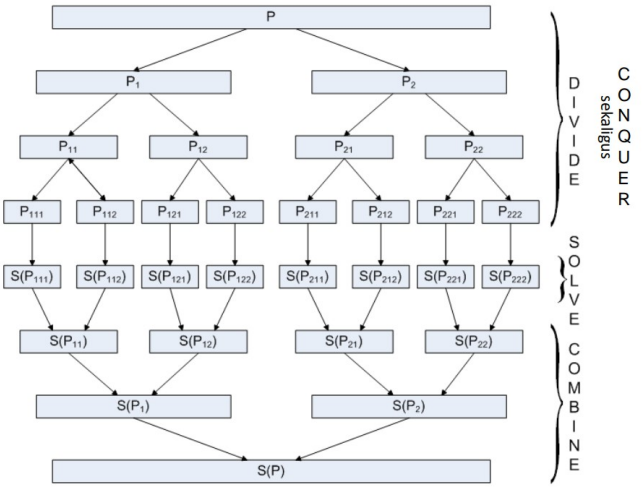


Fig. 1. Tiga tahap dalam algoritma *divide and conquer*
(Sumber: informatika.stei.itb.ac.id/~rinaldi.munir)

Algoritma *divide and conquer* sendiri dapat diparalelkan sehingga tahap penyelesaian subpersoalan dapat dilakukan secara *concurrent*. Hal tersebut dapat meningkatkan kecepatan komputasi dengan semakin berkembangnya *multithreading* pada komputer. *Divide and conquer* juga memiliki banyak aplikasi dalam bidang komputasi. Beberapa algoritma yang menggunakan teknik *divide and conquer* diantaranya dalah *quicksort*, *mergesort*, dan *binary search*. Pendekatan *divide and conquer* memberikan pendekatan yang kuat untuk mendesain sebuah algoritma yang efisien.

B. Metode Bagging

Metode *bagging*, yang merupakan singkatan dari *bootstrap aggregating*, merupakan teknik *ensemble learning* yang umum digunakan dalam pembuatan model pembelajaran mesin. Cara kerja metode ini adalah dengan menggabungkan beberapa model yang dilatih pada subset-subset dari dataset yang diberikan.

Pada awalnya, metode ini membuat sampel *bootstrap* dari dataset awal. Sampel diambil dengan memilih subset dari dataset awal dengan pengembalian. Hal tersebut membuat sebuah subset dapat memiliki data yang terduplikasi. Dengan membuat sampel *bootstrap*, dihasilkan dataset pelathian yang terdiversifikasi yang dapat memunculkan aspek-aspek baru dari data dan menambah variansi data pada model.

Tahap berikutnya adalah melatih model pada setiap sampel secara independen. Model yang dapat digunakan adalah model apapun seperti pohon keputusan, *random forest*, atau *support vector machines*. Setiap model dilatih pada subset dataset yang berbeda sehingga proses pelatihan akan menjadi lebih beragam. Tujuan dari metode ini adalah variansi prediksi dapat dikurangi dan performa prediksi dapat ditingkatkan dengan menggabungkan prediksi dari beberapa model.

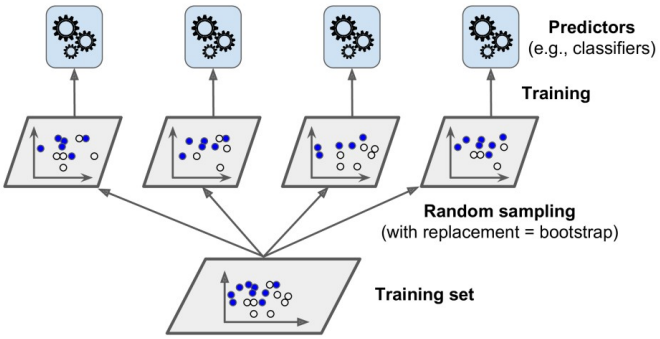


Fig. 2. Metode *bagging* sebagai implementasi *divide and conquer* dalam pembelajaran mesin
(Sumber: *Hands-on Machine Learning*, 2019)

Pada tahap prediksi, metode *bagging* menggabungkan prediksi dari semua model dasar yang sudah dilatih. Penggabungan dapat dilakukan melalui skema voting mayoritas untuk persoalan klasifikasi, dimana kelas dengan suara terbanyak akan dijadikan sebagai prediksi akhir. Sementara untuk persoalan regresi, penggabungan dapat dilakukan dengan mengambil mean atau median dari prediksi model dasar. Dengan ini, model cenderung lebih tahan terhadap *overfitting* daripada model dasar.

Keuntungan dari metode *bagging* sendiri adalah dapat meningkatkan generalisasi dari sebuah model. Dengan melatih model dari subset-subset yang berbeda, metode *bagging* mengurangi dampak dari *outlier* yang dapat mempengaruhi model dasar. Selain itu, metode ini juga membantu model menangkap pola-pola dan hubungan-hubungan yang terdapat pada data sehingga prediksi yang dibuat dapat lebih akurat.

Metode ini sudah efektif digunakan dalam berbagai persoalan pembelajaran mesin dan telah menunjukkan berbagai peningkatan dalam hal performa. Metode ini menjadi sangat efektif saat model dasar yang digunakan merupakan *weak learner*. *Bagging* dapat meningkatkan performa sebuah model *weak learner* dengan mengurangi bias dan menggabungkan kelebihan dari setiap model. Contoh algoritma pelatihan yang menggunakan *bagging* adalah *random forest* dan *bagged neural networks*.

III. Metode Pengujian

Metode *bagging*, merupakan salah satu implementasi dari algoritma *divide and conquer* dalam pembelajaran mesin. *Dalam* makalah ini, akan dilakukan pengujian terhadap efektivitas metode *bagging* dalam meningkatkan performa sebuah model. Pengujian memanfaatkan beberapa *tools* dan *library* dari bahasa pemrograman Python. *Library* Scikit-learn digunakan untuk menghasilkan model yang digunakan sebagai pembanding. *Library* Pandas dan Numpy digunakan untuk memproses data yang akan digunakan.

A. Dataset yang Digunakan

Dataset yang digunakan dalam pengujian adalah dataset Titanic. Dataset ini berisi data penumpang kapal Titanic yang tenggelam pada tahun 1912. Dataset Titanic umum digunakan untuk melakukan prediksi apakah seorang penumpang berhasil selamat dari tragedi tersebut. Dataset ini dapat diakses pada https://www.kaggle.com/competitions/titanic/data.

Dataset Titanic berisi 891 data dan 12 fitur. Fitur-fitur pada dataset ini terbagi menjadi fitur numerik dan kategorikal. Tipe data serta persentase nilai kosong dari setiap fitur dapat dilihat pada Fig. 3. Deskripsi dari setiap fitur dapat dilihat pada Fig. 4.

Fitur	*Tipe Data*	*Nilai Kosong (%)*
PassengerId	int64	0.0
Survived	int64	0.0
Pclass	int64	0.0
Name	object	0.0
Sex	object	0.0
Age	float64	19.9
SibSp	int64	0.0
Parch	int64	0.0
Ticket	object	0.0
Fare	object	0.0
Cabin	object	77.1
Embarked	object	0.0

Fig. 3. Tabel tipe data dan persentase nilai kosong setiap fitur pada dataset

Fitur	*Deskripsi Fitur*
PassengerId	Indeks penumpang pada dataset
Survived	Fitur biner yang bernilai 1 jika penumpang selamat dan bernilai 0 jika penumpang tidak selamat
Pclass	Kelas dari setiap penumpang yang bernilai 1, 2, 3
Name	Nama lengkap penumpang beserta gelar/julukan yang dimilikinya
Sex	Jenis kelamin penumpang
Age	Umur penumpang
SibSp	Jumlah saudara dan sepupu yang dimiliki oleh penumpang
Parch	Jumlah orang tua dan anak yang dimiliki oleh penumpang
Ticket	Tiket yang dimiliki penumpang
Fare	Harga tiket penumpang
Cabin	Kode kabin yang ditempati penumpang
Embarked	Nama pelabuhan asal penumpang

Fig. 4. Tabel deskripsi setiap fitur pada dataset

B. Pemrosesan Awal Data

Sebelum data digunakan untuk pelatihan model, dataset diproses terlebih dahulu menggunakan beberapa metode pemrosesan data. Metode pemrosesan yang digunakan hanya meliputi metode pemrosesan dasar. Metode pemrosesan kompleks seperti *imputation*, *resampling*, dan *scaling* tidak dilakukan karena pengujian hanya akan digunakan untuk mengukur peningkatan performa model yang disebabkan metode *bagging*.

Beberapa fitur nonnumerik dihilangkan dari dataset karena model pembelajaran mesin hanya dapat belajar dari data-data numerik. Metode *encoding* untuk fitur kategorikal tidak dilakukan karena metode tersebut digunakan untuk meningkatkan performa model dan tidak diperlukan untuk pengujian ini. Fitur-fitur yang dihilangkan adalah PassengerId, Ticket, Cabin, Name, Sex, Embarked.

Pada tahap ini, fitur dan label dari dataset dipisahkan. Label yang digunakan adalah kolom Survived karena model akan digunakan untuk memprediksi apakah penumpang selamat atau tidak. Dataset sendiri dibagi menjadi dua: dataset untuk pelatihan model dan dataset untuk pengujian. Rasio dataset pelatihan dan pengujian yang digunakan adalah 80:20.

C. Pengukuran Kinerja Model

Kinerja model yang dihasilkan akan diukur menggunakan metrik akurasi dan *receiver operating characteristic* (ROC). Nilai akurasi sebuah model dapat diukur dengan persamaan,

$$akurasi = \frac{prediksi\,benar}{total\,data}$$

Akurasi dari sebuah model merupakan metrik dasar untuk mengukur kemampuan prediksi model.

Reciever operating characteristic (ROC) merupakan metrik yang dapat mengukur kinerja model dengan lebih detail. Perhitungan nilai ROC melibatkan nilai *true positive*, *false positive*, *true negative*, dan *false negative*. Keempat nilai tersebut dapat

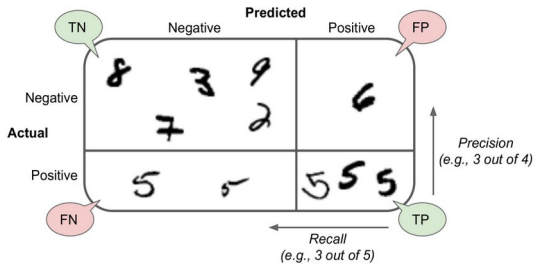


Fig. 5. Nilai *true positive*, *false positive*, *true negative*, dan *false negative* untuk klasifikasi angka 5 dari dataset gambar
(Sumber : *Hands-on Machine Learning*, 2019)

Keempat nilai tersebut kemudian digunakan untuk menghitung *true positive rate* (TPR) dan *false positive rate* (FPR). TPR dapat dihitung dengan persamaan,

$$TPR = \frac{True\ Positve}{True\ Positive + False\ Negative}$$

Sementara FPR dapat dihitung dengan persamaan,

$$FPR = \frac{False\ Positve}{False\ Positive + True\ Negative}$$

ROC merupakan kurva yang menggambarkan nilai *true positive rate* (TPR) terhadap *false positive rate* (FPR). Nilai ROC dapat dihitung dengan menghitung daerah di bawah kurva ROC. Nilai tersebut kemudian digunakan untuk mengukur performa model.

IV. Hasil dan Pembahasan

Berdasarkan pengujian yang dilakukan dapat terlihat bahwa metode *bagging* dapat meningkatkan performa model secara signifikan. Terdapat peningkatan akurasi sebesar 6.7% dan pengingkatan nilai ROC sebesar 9.2%. Berikut ini merupakan kode program yang digunakan untuk membuat model.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import BaggingClassifier

tree_clf = DecisionTreeClassifier()
tree_clf.fit(X_train, y_train)

bag_clf = BaggingClassifier(
    DecisionTreeClassifier(), n_estimators=100,
    max_samples=100, bootstrap=True, n_jobs=-1
)

bag_clf.fit(X_train, y_train)
```

Fig. 6. Model pohon keputusan beserta model yang menggunakan metode *bagging*
(Sumber: Dokumentasi pribadi)

Nilai akurasi prediksi diuji menggunakan *library* Scikit-learn dan didapatkan hasil sebagai berikut.

```
from sklearn.metrics import accuracy_score

y_pred = tree_clf.predict(X_test)
accuracy_score(y_test, y_pred)
```

```
0.659217877094972
```

Fig. 7. Nilai akurasi model dasar pohon keputusan
(Sumber: Dokumentasi pribadi)

```
y_pred = bag_clf.predict(X_test)
accuracy_score(y_test, y_pred)
```

```
0.7262569832402235
```

Fig. 8. Nilai akurasi model pohon keputusan yang menggunakan metode *bagging*
(Sumber: Dokumentasi pribadi)

Selain akurasi, performa model dapat dikukur menggunakan nilai ROC. Kurva ROC sendiri berguna untuk memvisualisasikan kinerja model. Berikut merupakan kode program untuk menampilkan kurva ROC beserta kurva hasil untuk kedua model yang diuji.

```
import matplotlib.pyplot as plt

def plot_roc_curve(fpr, tpr, label=None):
    plt.plot(fpr, tpr, linewidth=2, label=label)
    plt.plot([0, 1], [0, 1], "k--")
    plt.grid()
    plt.title("ROC Curve")
    plt.xlabel("False Positive Rate")
    plt.ylabel("True Positive Rate (Recall)")
```

Fig. 9. Fungsi untuk menampilkan kurva ROC

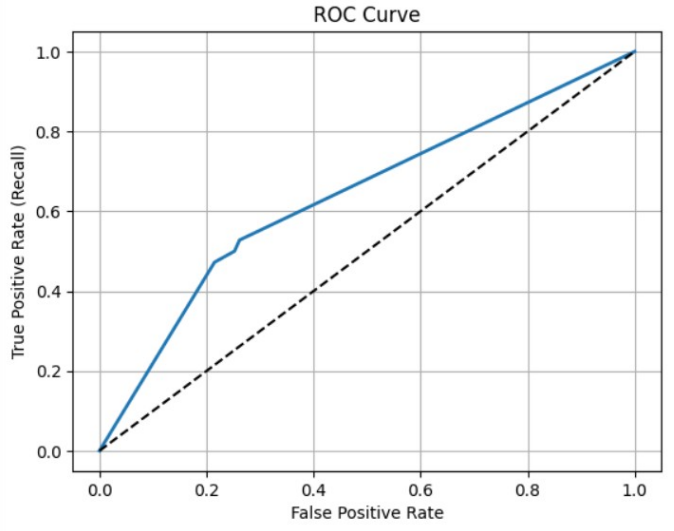


Fig. 10. Kurva ROC untuk model dasar pohon keputusan
(Sumbe: Dokumentasi pribadi)

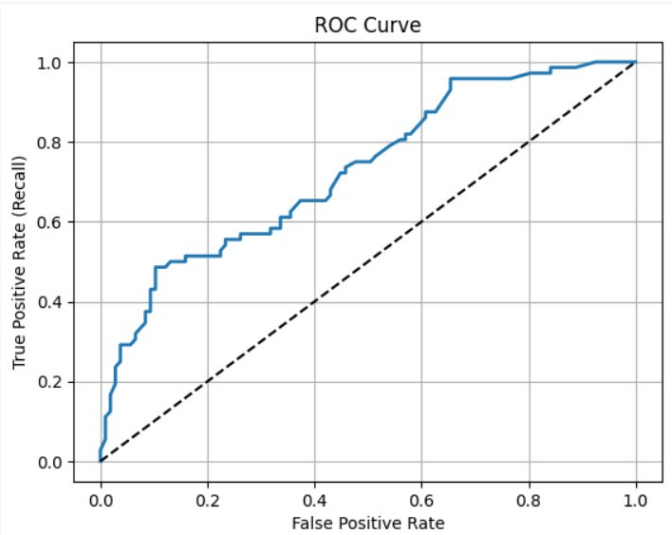


Fig. 11. Kurva ROC untuk model pohon keputusan yang menggunakan metode *bagging*
(Sumber: Dokumentasi pribadi)

Berdasarkan metrik pengujian yang digunakan, metode *bagging* berhasil meningkatkan performa dari sebuah model. Peningkatan performa yang relatif tinggi hanya dengan menambah sebuah teknik menunjukkan efektivitas dari metode *bagging*. Efektivitas tersebut berasal dari penggunaan algoritma *divide and conquer* yang efisien dalam persoalan pembelajaran mesin.

V. KESIMPULAN

Pada makalah ini telah dilakukan pengujian untuk menunjukkan efektivitas algoritma *divide and conquer* melalui metode *bagging* dalam meningkatkan akurasi sebuah model pembelajaran mesin. Pendekatan *divide and conquer* melalui metode *bagging* mampu meningkatkan performa model dengan memecah persoalan *ensemble learning* yang kompleks menjadi persoalan yang lebih sederhana. Metode *bagging* membuat sebuah model menangkap aspek-aspek baru dari sebuah dataset. Hal tersebut meningkatkan kemampuan generalisasi sebuah model dan mengurangi efek *overfitting*.

Keberhasilan tersebut terlihat dengan meningkatnya nilai akurasi dan ROC yang digunakan sebagai metrik pengukuran sebuah model. Efektivitas metode *bagging* membuat metode ini sangat cocok untuk digunakan dalam persoalan-persoalan pembelajaran mesin lainnya.

REFERNSI

[1] Munir, Rinaldi, 2021. *Algoritma Divide and Conquer (Bagian 1)*. Dilansir dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag1.pdf.

[2] Munir, Rinaldi, 2021. *Algoritma Divide and Conquer (Bagian 2)*. Dilansir dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag2.pdf.

[3] Munir, Rinaldi, 2021. *Algoritma Divide and Conquer (Bagian 3)*. Dilansir dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag3.pdf.

[4] Munir, Rinaldi, 2021. *Algoritma Divide and Conquer (Bagian 4)*. Dilansir dari https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Greedy-(2021)-Bag4.pdf.

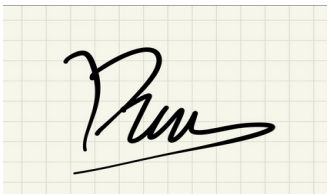
[5] Geron, Aurelien, 2019. *Hands-on Machine Learning.*

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 22 Mei 2023

Ttd



Rava Maulana - 13521149